Analyze A/B Test Results

Table of Contents

Introduction

A/B tests are very commonly performed by data analysts and data scientists. It is important that you get some practice working with the difficulties of these

Part I - Probability

To get started, let's import our libraries.

In [11]:
```
import pandas as pd
import numpy as np
import random
import matplotlib.pyplot as plt
import statsmodels.api as sm
%matplotlib inline
#We are setting the seed to assure you get the same answers on quizzes as we set up
random.seed(42)
```

`1.` Now, read in the `ab_data.csv` data. Store it in `df`. **Use your dataframe to answer the questions in Quiz 1 of the classroom.**

a. Read in the dataset and take a look at the top few rows here:

In [12]:
```
df=pd.read_csv(r"C:\Users\HP\.jupyter\ab_data.csv")
df.head()
```

Out[12]:

	user_id	timestamp	group	landing_page	converted
0	851104	2017-01-21 22:11:48.556739	control	old_page	0
1	804228	2017-01-12 08:01:45.159739	control	old_page	0
2	661590	2017-01-11 16:55:06.154213	treatment	new_page	0
3	853541	2017-01-08 18:28:03.143765	treatment	new_page	0
4	864975	2017-01-21 01:52:26.210827	control	old_page	1

b. Use the below cell to find the number of rows in the dataset.

In [13]:
```
df.shape
```

Out[13]:
```
(294478, 5)
```

c. The number of unique users in the dataset.

In [14]:
```
df.user_id.nunique()
```
Out[14]:
```
290584
```

d. The proportion of users converted.

In [15]:
```
df.converted.mean()
```
Out[15]:
```
0.11965919355605512
```

e. The number of times the `new_page` and `treatment` don't line up.

In [16]:
```
df.query("(group=='treatment' and landing_page!='new_page')or(group!='treatment' and lan
```
Out[16]:
```
user_id         3893
timestamp       3893
group           3893
landing_page    3893
converted       3893
dtype: int64
```

f. Do any of the rows have missing values?

In [17]:
```
df.isnull()
```
Out[17]:

	user_id	timestamp	group	landing_page	converted
0	False	False	False	False	False
1	False	False	False	False	False
2	False	False	False	False	False
3	False	False	False	False	False
4	False	False	False	False	False
...	...	...	...	...	...
294473	False	False	False	False	False
294474	False	False	False	False	False
294475	False	False	False	False	False
294476	False	False	False	False	False
294477	False	False	False	False	False

294478 rows × 5 columns

`2.` For the rows where **treatment** is not aligned with **new_page** or **control** is not aligned with **old_page**, we cannot be sure if this row truly received the new or old page. Use **Quiz 2** in the classroom to provide how we should handle these rows.

a. Now use the answer to the quiz to create a new dataset that meets the specifications from the quiz. Store your new dataframe in **df2**.

In [18]:
```
df2=df.query("(group=='control' and landing_page=='old_page')or(group=='treatment' and l
df2
```
Out[18]:

	user_id	timestamp	group	landing_page	converted

0	851104	2017-01-21 22:11:48.556739	control	old_page	0
1	804228	2017-01-12 08:01:45.159739	control	old_page	0
2	661590	2017-01-11 16:55:06.154213	treatment	new_page	0
3	853541	2017-01-08 18:28:03.143765	treatment	new_page	0
4	864975	2017-01-21 01:52:26.210827	control	old_page	1
...	...	...	...	...	...
294473	751197	2017-01-03 22:28:38.630509	control	old_page	0
294474	945152	2017-01-12 00:51:57.078372	control	old_page	0
294475	734608	2017-01-22 11:45:03.439544	control	old_page	0
294476	697314	2017-01-15 01:20:28.957438	control	old_page	0
294477	715931	2017-01-16 12:40:24.467417	treatment	new_page	0

290585 rows × 5 columns

In [19]:
```
# Double Check all of the correct rows were removed - this should be 0
df2[((df2['group'] == 'treatment') == (df2['landing_page'] == 'new_page')) == False].sha
```

Out[19]:
```
0
```

`3.` Use **df2** and the cells below to answer questions for **Quiz3** in the classroom.

a. How many unique **user_id**s are in **df2**?

In [20]:
```
df2.user_id.nunique()
```

Out[20]:
```
290584
```

b. There is one **user_id** repeated in **df2**. What is it?

In [21]:
```
duplicate=df2[df2['user_id'].duplicated()]
duplicate.iloc[:,0]
```

Out[21]:
```
2893    773192
Name: user_id, dtype: int64
```

c. What is the row information for the repeat **user_id**?

In [22]:
```
duplicate=df2[df2['user_id'].duplicated()]
duplicate
```

Out[22]:

	user_id	timestamp	group	landing_page	converted
2893	773192	2017-01-14 02:55:59.590927	treatment	new_page	0

d. Remove **one** of the rows with a duplicate **user_id**, but keep your dataframe as **df2**.

In [23]:
```
df2.drop_duplicates(subset=['user_id'],inplace=True)
df2[df2['user_id'].duplicated()]
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_11112\4182538992.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
```

```
guide/indexing.html#returning-a-view-versus-a-copy
  df2.drop_duplicates(subset=['user_id'],inplace=True)
```

Out[23]:

user_id	timestamp	group	landing_page	converted

`4.` Use **df2** in the below cells to answer the quiz questions related to **Quiz 4** in the classroom.

a. What is the probability of an individual converting regardless of the page they receive?

In [24]:
```
pro_converted=df2.converted.mean()
pro_converted
```

Out[24]:
```
0.11959708724499628
```

b. Given that an individual was in the `control` group, what is the probability they converted?

In [25]:
```
pro_control=df2.query('group =="control"')['converted'].mean()
pro_control
```

Out[25]:
```
0.1203863045004612
```

c. Given that an individual was in the `treatment` group, what is the probability they converted?

In [26]:
```
pro_treatment=df2.query('group =="treatment"')['converted'].mean()
pro_treatment
```

Out[26]:
```
0.11880806551510564
```

In [27]:
```
# Calculate the actual difference (obs_diff) between the conversion rates for the two gr
obs_diff=pro_treatment-pro_control
obs_diff
```

Out[27]:
```
-0.0015782389853555567
```

d. What is the probability that an individual received the new page?

In [28]:
```
pro_new=df2.query('landing_page =="new_page"').shape[0]/df2.shape[0]
pro_new
```

Out[28]:
```
0.5000619442226688
```

e. Consider your results from a. through d. above, and explain below whether you think there is sufficient evidence to say that the new treatment page leads to more conversions.

The proportions are very close

Part II - A/B Test

Notice that because of the time stamp associated with each event, you could technically run a hypothesis test continuously as each observation was observed.

However, then the hard question is do you stop as soon as one page is considered significantly better than another or does it need to happen consistently for a certain amount of time? How long do you run to render a decision that neither page is better than another?

These questions are the difficult parts associated with A/B tests in general.

`1.` For now, consider you need to make the decision just based on all the data provided. If you want to assume that the old page is better unless the new page proves to be definitely better at a Type I error rate of 5%, what should your null and alternative hypotheses be? You can state your hypothesis in terms of words or in terms of p_{old} and p_{new}, which are the converted rates for the old and new pages.

We need more than a month to get a correct result.

`2.` Assume under the null hypothesis, p_{new} and p_{old} both have "true" success rates equal to the **converted** success rate regardless of page - that is p_{new} and p_{old} are equal. Furthermore, assume they are equal to the **converted** rate in **ab_data.csv** regardless of the page.

Use a sample size for each page equal to the ones in **ab_data.csv**.

Perform the sampling distribution for the difference in **converted** between the two pages over 10,000 iterations of calculating an estimate from the null.

Use the cells below to provide the necessary parts of this simulation. If this doesn't make complete sense right now, don't worry - you are going to work through the problems below to complete this problem. You can use **Quiz 5** in the classroom to make sure you are on the right track.

a. What is the **convert rate** for p_{new} under the null?

In [29]:
```
convert_rate=df2.converted.mean()
convert_rate
```

Out[29]:
```
0.11959708724499628
```

b. What is the **convert rate** for p_{old} under the null?

In [30]:
```
convert_rate=len(df2.query('converted==1'))/len(df2.query("converted==1 or converted==0"
convert_rate
```

Out[30]:
```
0.11959708724499628
```

c. What is n_{new}?

In [31]:
```
n_new=len(df2[df2['landing_page']=='new_page'])
n_new
```

Out[31]:
```
145310
```

d. What is n_{old}?

In [32]:
```
n_old=len(df2[df2['landing_page']=='old_page'])
n_old
```

Out[32]:
```
145274
```

e. Simulate n_{new} transactions with a convert rate of p_{new} under the null. Store these n_{new} 1's and 0's in

new_page_converted.

In [33]:
```
new_page_converted=np.random.choice([0,1],n_new,p=[(1-convert_rate),convert_rate])
new_page_converted
```

Out[33]:
```
array([0, 0, 0, ..., 0, 0, 0])
```

f. Simulate n_{old} transactions with a convert rate of p_{old} under the null. Store these n_{old} 1's and 0's in **old_page_converted**.

In [34]:
```
old_page_converted=np.random.choice([0,1],n_new,p=[(1-convert_rate),convert_rate])
old_page_converted
```

Out[34]:
```
array([0, 0, 0, ..., 0, 0, 0])
```

g. Find p_{new} - p_{old} for your simulated values from part (e) and (f).

In [35]:
```
diff=new_page_converted.mean()-old_page_converted.mean()
diff
```

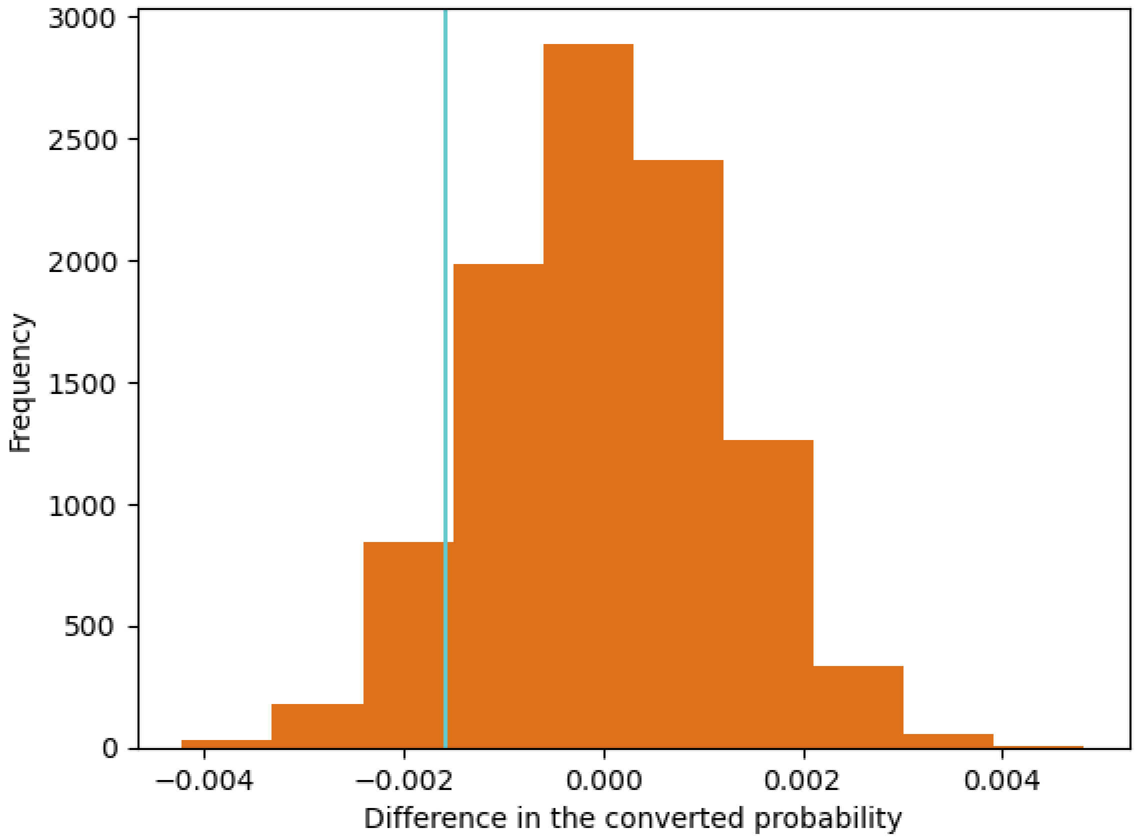
Out[35]:
```
0.0015346500584956374
```

h. Simulate 10,000 p_{new} - p_{old} values using this same process similarly to the one you calculated in parts **a. through g.** above. Store all 10,000 values in a numpy array called **p_diffs**.

In [76]:
```
p_diffs=[]
for _ in range(10000):
    new_page_converted=np.random.choice([0,1],n_new,p=[(1-convert_rate),convert_rate])
    old_page_converted=np.random.choice([0,1],n_old,p=[(1-convert_rate),convert_rate])
    diff=new_page_converted.mean()-old_page_converted.mean()
    p_diffs.append(diff)
```

i. Plot a histogram of the **p_diffs**. Does this plot look like what you expected? Use the matching problem in the classroom to assure you fully understand what was computed here.

In [86]:
```
plt.hist(p_diffs,color='#DF711B')
plt.xlabel('Difference in the converted probability')
plt.ylabel("Frequency")
plt.axvline(obs_diff,color="#64C9CF");
```



j. What proportion of the **p_diffs** are greater than the actual difference observed in **ab_data.csv**?

In [38]:
```
p_diffs=np.array(p_diffs)
(p_diffs>obs_diff).mean()
```

Out[38]:
```
0.9035
```

In [39]:
```
p_diffs=p_diffs.mean()
p_diffs
```

Out[39]:
```
-7.815919195166343e-06
```

k. In words, explain what you just computed in part **j.** What is this value called in scientific studies? What does this value mean in terms of whether or not there is a difference between the new and old pages?

This value 0.9038 is p-value, this value higher of Type I error rate (0.05), that means the null hypothesis is not rejected and the new page not higher conversion rates

l. We could also use a built-in to achieve similar results. Though using the built-in might be easier to code, the above portions are a walkthrough of the ideas that are critical to correctly thinking about statistical significance. Fill in the below to calculate the number of conversions for each page, as well as the number of individuals who received each page. Let `n_old` and `n_new` refer the the number of rows associated with the old page and new pages, respectively.

In [40]:
```
import statsmodels.api as sm

# number of conversions with the old_page
convert_old = len(df2.query('landing_page=="old_page" & converted == 1'))

# number of conversions with the new_page
```

```
convert_new = len(df2.query('landing_page=="new_page" & converted == 1'))

# number of individuals who were shown the old_page
n_old = len(df2[df2['landing_page']=='old_page'])

# number of individuals who received new_page
n_new = len(df2[df2['landing_page']=='new_page'])
```

m. Now use `stats.proportions_ztest` to compute your test statistic and p-value. Here is a helpful link on using the built in.

In [41]:
```
z_score,p_value=sm.stats.proportions_ztest([convert_new,convert_old],[n_new,n_old],alter
print(z_score,',',p_value)
```

```
-1.3109241984234394 , 0.9050583127590245
```

n. What do the z-score and p-value you computed in the previous question mean for the conversion rates of the old and new pages? Do they agree with the findings in parts **j.** and **k.**?

The z-score = -1.31 means we are very close from mean difference in the converted probability -1.3521582055095966e-05 and p-value mean there is no statstically sinficant to reject the null hypothesis

Part III - A regression approach

`1.` In this final part, you will see that the result you acheived in the previous A/B test can also be acheived by performing regression.

a. Since each row is either a conversion or no conversion, what type of regression should you be performing in this case?

Logistic Regression.

b. The goal is to use **statsmodels** to fit the regression model you specified in part **a.** to see if there is a significant difference in conversion based on which page a customer receives. However, you first need to create a column for the intercept, and create a dummy variable column for which page each user received. Add an **intercept** column, as well as an **ab_page** column, which is 1 when an individual receives the **treatment** and 0 if **control**.

In [42]:
```
df2
```

Out[42]:

	user_id	timestamp	group	landing_page	converted
0	851104	2017-01-21 22:11:48.556739	control	old_page	0
1	804228	2017-01-12 08:01:45.159739	control	old_page	0
2	661590	2017-01-11 16:55:06.154213	treatment	new_page	0
3	853541	2017-01-08 18:28:03.143765	treatment	new_page	0
4	864975	2017-01-21 01:52:26.210827	control	old_page	1
...	...	...	...	...	...
294473	751197	2017-01-03 22:28:38.630509	control	old_page	0

294474	945152	2017-01-12 00:51:57.078372	control	old_page	0
294475	734608	2017-01-22 11:45:03.439544	control	old_page	0
294476	697314	2017-01-15 01:20:28.957438	control	old_page	0
294477	715931	2017-01-16 12:40:24.467417	treatment	new_page	0

290584 rows × 5 columns

In [43]:
```
df2['intercept']=1
df2['ab_page']=pd.get_dummies(df2['group'])['treatment']
df2
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_11112\3682375604.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
  df2['intercept']=1
C:\Users\HP\AppData\Local\Temp\ipykernel_11112\3682375604.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
  df2['ab_page']=pd.get_dummies(df2['group'])['treatment']
```

Out[43]:

	user_id	timestamp	group	landing_page	converted	intercept	ab_page
0	851104	2017-01-21 22:11:48.556739	control	old_page	0	1	0
1	804228	2017-01-12 08:01:45.159739	control	old_page	0	1	0
2	661590	2017-01-11 16:55:06.154213	treatment	new_page	0	1	1
3	853541	2017-01-08 18:28:03.143765	treatment	new_page	0	1	1
4	864975	2017-01-21 01:52:26.210827	control	old_page	1	1	0
...	...	...	...	...	...	...	...
294473	751197	2017-01-03 22:28:38.630509	control	old_page	0	1	0
294474	945152	2017-01-12 00:51:57.078372	control	old_page	0	1	0
294475	734608	2017-01-22 11:45:03.439544	control	old_page	0	1	0
294476	697314	2017-01-15 01:20:28.957438	control	old_page	0	1	0
294477	715931	2017-01-16 12:40:24.467417	treatment	new_page	0	1	1

290584 rows × 7 columns

c. Use **statsmodels** to import your regression model. Instantiate the model, and fit the model using the two columns you created in part **b.** to predict whether or not an individual converts.

In [44]:
```
log_model=sm.Logit(df2['converted'],df2[['intercept','ab_page']])
result=log_model.fit()
```

```
Optimization terminated successfully.
         Current function value: 0.366118
         Iterations 6
```

d. Provide the summary of your model below, and use it as necessary to answer the following questions.

In [45]:
```
result.summary2()
```

Out[45]:

Model:	Logit	Pseudo R-squared:	0.000
Dependent Variable:	converted	AIC:	212780.3502
Date:	2022-12-03 21:28	BIC:	212801.5095
No. Observations:	290584	Log-Likelihood:	-1.0639e+05
Df Model:	1	LL-Null:	-1.0639e+05
Df Residuals:	290582	LLR p-value:	0.18988
Converged:	1.0000	Scale:	1.0000
No. Iterations:	6.0000		

	Coef.	Std.Err.	z	P>|z|	[0.025	0.975]
intercept	-1.9888	0.0081	-246.6690	0.0000	-2.0046	-1.9730
ab_page	-0.0150	0.0114	-1.3109	0.1899	-0.0374	0.0074

In [46]:
```
#We exponentiate the ab_page coefficient to interpret it.
np.exp(-0.0150)
```

Out[46]:
```
0.9851119396030626
```

In [47]:
```
#When multiplicative changes are less than 1, it is usually useful to calculate the reci
1/np.exp(-0.0150)
```

Out[47]:
```
1.015113064615719
```

e. What is the p-value associated with **ab_page**? Why does it differ from the value you found in **Part II**?

Hint: What are the null and alternative hypotheses associated with your regression model, and how do they compare to the null and alternative hypotheses in the **Part II**?

The P-value in regression model is 0.1899 this value higher of Type I error rate (0.05).

Hypothesis in Part II is one-sided and in part III is two-sided.

f. Now, you are considering other things that might influence whether or not an individual converts. Discuss why it is a good idea to consider other factors to add into your regression model. Are there any disadvantages to adding additional terms into your regression model?

Yes, we can benefit from increasing factors, but they will take time and effort.

g. Now along with testing if the conversion rate changes for different pages, also add an effect based on which country a user lives. You will need to read in the **countries.csv** dataset and merge together your datasets on the appropriate rows. Here are the docs for joining tables.

Does it appear that country had an impact on conversion? Don't forget to create dummy variables for these country columns - **Hint: You will need two columns for the three dummy variables.** Provide the statistical output as well as a written response to answer this question.

In [87]:
```
df_countries=pd.read_csv(r"C:\Users\HP\.jupyter\countries.csv",index_col='user_id')
```

In [49]:
```
df_merged=df2.set_index('user_id').join(df_countries,how='inner')
df_merged
```

Out[49]:

user_id	timestamp	group	landing_page	converted	intercept	ab_page	country
851104	2017-01-21 22:11:48.556739	control	old_page	0	1	0	US
804228	2017-01-12 08:01:45.159739	control	old_page	0	1	0	US
661590	2017-01-11 16:55:06.154213	treatment	new_page	0	1	1	US
853541	2017-01-08 18:28:03.143765	treatment	new_page	0	1	1	US
864975	2017-01-21 01:52:26.210827	control	old_page	1	1	0	US
...	...	...	...	...	...	...	...
751197	2017-01-03 22:28:38.630509	control	old_page	0	1	0	US
945152	2017-01-12 00:51:57.078372	control	old_page	0	1	0	US
734608	2017-01-22 11:45:03.439544	control	old_page	0	1	0	US
697314	2017-01-15 01:20:28.957438	control	old_page	0	1	0	US
715931	2017-01-16 12:40:24.467417	treatment	new_page	0	1	1	UK

290584 rows × 7 columns

In [54]:
```
### Create the necessary dummy variables
df_merged[['UK', 'US', 'CA']]=pd.get_dummies(df_merged['country'])
df_merged
```

Out[54]:

user_id	timestamp	group	landing_page	converted	intercept	ab_page	country	US	UK	CA
851104	2017-01-21 22:11:48.556739	control	old_page	0	1	0	US	0	0	1
804228	2017-01-12 08:01:45.159739	control	old_page	0	1	0	US	0	0	1
661590	2017-01-11 16:55:06.154213	treatment	new_page	0	1	1	US	0	0	1
853541	2017-01-08 18:28:03.143765	treatment	new_page	0	1	1	US	0	0	1
864975	2017-01-21 01:52:26.210827	control	old_page	1	1	0	US	0	0	1
...	...	...	...	...	...	...	...	...	...	...
751197	2017-01-03 22:28:38.630509	control	old_page	0	1	0	US	0	0	1
945152	2017-01-12 00:51:57.078372	control	old_page	0	1	0	US	0	0	1
734608	2017-01-22 11:45:03.439544	control	old_page	0	1	0	US	0	0	1

697314	2017-01-15 01:20:28.957438	control	old_page	0	1	0	US	0	0	1
715931	2017-01-16 12:40:24.467417	treatment	new_page	0	1	1	UK	1	0	0

290584 rows × 10 columns

In [64]:
```
log_model1=sm.Logit(df_merged['converted'],df_merged[['intercept','ab_page','US','CA']])
result1=log_model1.fit()
result1.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.366113
         Iterations 6
```

Out[64]:

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	290584
Model:	Logit	**Df Residuals:**	290580
Method:	MLE	**Df Model:**	3
Date:	Sat, 03 Dec 2022	**Pseudo R-squ.:**	2.323e-05
Time:	21:56:18	**Log-Likelihood:**	-1.0639e+05
converged:	True	**LL-Null:**	-1.0639e+05
Covariance Type:	nonrobust	**LLR p-value:**	0.1760

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
intercept	-2.0300	0.027	-76.249	0.000	-2.082	-1.978
ab_page	-0.0149	0.011	-1.307	0.191	-0.037	0.007
US	0.0506	0.028	1.784	0.074	-0.005	0.106
CA	0.0408	0.027	1.516	0.130	-0.012	0.093

h. Though you have now looked at the individual factors of country and page on conversion, we would now like to look at an interaction between page and country to see if there significant effects on conversion. Create the necessary additional columns, and fit the new model.

Provide the summary results, and your conclusions based on the results.

In [66]:
```
df_merged['US_ab_page']=df_merged['ab_page']*df_merged['US']
df_merged['CA_ab_page']=df_merged['ab_page']*df_merged['CA']
df_merged
```

In [69]:
```
### Fit Your Linear Model And Obtain the Results
log_model2=sm.Logit(df_merged['converted'],df_merged[['intercept','ab_page','US','CA','U
result2=log_model2.fit()
result2.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.366109
         Iterations 6
```

Out[69]:

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	290584
Model:	Logit	**Df Residuals:**	290578
Method:	MLE	**Df Model:**	5

Date:	Sat, 03 Dec 2022	**Pseudo R-squ.:**	3.482e-05
Time:	22:06:41	**Log-Likelihood:**	-1.0639e+05
converged:	True	**LL-Null:**	-1.0639e+05
Covariance Type:	nonrobust	**LLR p-value:**	0.1920

	coef	std err	z	P>|z|	[0.025	0.975]
intercept	-2.0040	0.036	-55.008	0.000	-2.075	-1.933
ab_page	-0.0674	0.052	-1.297	0.195	-0.169	0.034
US	0.0118	0.040	0.296	0.767	-0.066	0.090
CA	0.0175	0.038	0.465	0.642	-0.056	0.091
US_ab_page	0.0783	0.057	1.378	0.168	-0.033	0.190
CA_ab_page	0.0469	0.054	0.872	0.383	-0.059	0.152

Conclusions

Based on the summary in regression model, we failed to reject the null hypothesis. It was all the greater than Type I error rate (0.05). There are no effect of page and country to predict the conversion.